ACM JAVALAB 11

Random-Access Files

Java provides the RandomAccessFile class to allow data to be read from and
written to at any locations in the file.

All of the streams you have used so far are known as read-only or write-only streams.
These
streams are called sequential streams. A file that is opened using a sequential stream is
called
a sequential-access file.

The contents of a sequential-access file cannot be updated. However, it is often
necessary to modify files.

Java provides the **RandomAccessFile** class to allow data to be read from and written to
at any locations in a file. A file that is opened using the **RandomAccessFile** class is
known as a
random-access file.

When creating a RandomAccessFile, you can specify one of two modes: **r** or **rw**. Mode
r means that the stream is **read-only**, and mode **rw** indicates that the stream **allows
both read
and write.**

For example, the following statement creates a new stream, raf, that allows the
program to read from and write to the file test.txt:

```
RandomAccessFile raf = new RandomAccessFile("test.txt", "rw");
```

If test.txt already exists, raf is created to access it; if test.txt does not exist, a new file
named
test.txtis created, and raf is created to access the new file.

The method **raf.length()** returns the number of bytes in test.txt any given time. If you
append new data into the file, **raf.length()** increases.

For a **RandomAccessFile** raf, you can use the **raf.seek(position)** method to move
the file pointer to a specified position. **raf.seek(0)** moves it to the beginning of the file,
and **raf.seek(raf.length())** moves it to the end of the file.

If you read an int value using **read(4)**, the JVM reads 4 bytes from the file pointer, and now the file pointer is 4 bytes ahead of the previous location.

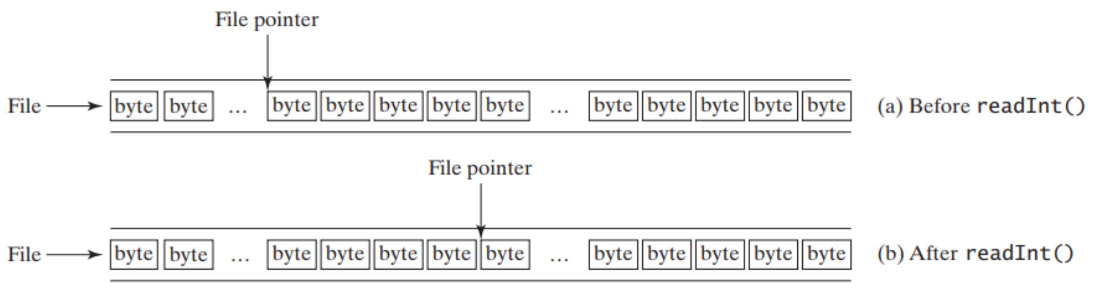
File pointer
File
byte byte ... byte byte byte byte byte ... byte byte byte byte byte
(a) Before readInt()
File pointer
File
byte byte ... byte byte byte byte byte ... byte byte byte byte byte
(b) After readInt()

FIGURE 17.19 After an int value is read, the file pointer is moved 4 bytes ahead.

```
package lab10;

import java.io.IOException;
import java.io.RandomAccessFile;

public class RandomAccessFileExample {


  public static void main(String[] args) {
    String FILE = "data.txt";

    try {
      System.out.println(new String(readSomeDataFromFile(FILE, 23, 34)));
      writeDataToFile(FILE, "\nWe are coding JAVA",85);
    } catch (Exception e) {
      System.out.println("Error");
    }
  }

  private static byte[] readSomeDataFromFile(String file, int pos, int size) throws IOException
  {
    RandomAccessFile file_access = new RandomAccessFile(file, "r");
    file_access.seek(pos);
    System.out.println(file_access.length());
    byte[] bytesToRead = new byte[size];
    file_access.read(bytesToRead);

    file_access.close();

    return bytesToRead;
  }

  private static void writeDataToFile(String file, String data, int pos) throws IOException
  {
    //we are giving both reading and writing permission
    RandomAccessFile file_access = new RandomAccessFile(file, "rw");

    file_access.seek(pos);
    file_access.write(data.getBytes());
    file_access.close();
  }
}
```